2009 Seventh Annual Communication Networks and Services Research Conference

E-health Services with Secure Mobile Agent

Rossilawati Sulaiman, Xu Huang, Dharmendra Sharma
Department of Information Science & Engineering
University of Canberra
Australia
{Rossillawati.Sulaiman, Xu.Huang, Dharmendra.Sharma}@canberra.edu.au

Abstract— **This paper establishes a new security mechanism for online communications using mobile agents. E-health has been widely used as a communication system and information changes to deliver health services to the users through the Internet. However, the Internet imposes threats that will risk the information in transit. Therefore it is important to ensure that the communication is safe and the user's privacy is protected. This paper extends our previous research results, Multilayer Communication (MLC) approach [1], and introduced mobile agents to MLC in e-health. The cryptographic protocols such as encryption/decryption, digital signature, and hash code are used as protection mechanisms. We focus on how Sender can securely transfer sensitive information to Recipient while still maintaining control over it. Sender keeps the key for decryption at his/her side until the agent needs it. A token is carried by the agent, which is used as a mean to call home platform to obtain the key for decryption processes. The significant of this new mechanism is that it gives Sender control over the transmitted data. Recipient can also avoid burden with the encryption details as long as he/she knows that the message is indeed comes from Recipient.**

Keywords-Mobile agent, security, e-health.

I. INTRODUCTION

The electronic health, or e-health, enables the delivery of health care services through the Internet. E-health improves the relationship between patients and doctors, as well as provides online education for patients and physicians through online resources [2]. In e-health, patients and medical experts communicate by exchanging information. This is useful especially for remote users. For example, a patient at home can communicate remotely with a doctor at a hospital. Both parties can communicate online, for consultation sessions or discuss anything related to health care issues. Currently, many applications have been developed to support communications and information exchanges among doctors, and patients, such as video conferencing [3], emails [4], and web-based messaging [5],[6]. These applications involve the use of the Internet network. However, there exist many threats to the Internet, such as network attacks and information breaches by intruders, or malicious software, which is a program that is purposely created to perform illegal operation on the computer system, like viruses and worms. These threats can cause severe damages to the computer systems as well as the information. The information might be stolen or modified and may cause undesirable consequences. For example, if a patient's medical information is violated or modified by a third party, the patient may not receive correct medications and furthermore, may endanger the patient's life. Therefore, it is important to secure these communications.

In this research, emphasis is given on securing online communications, specifically in the e-health domain using mobile agents [7] and cryptography protocols. The mobile agents are chosen to carry sensitive information during a communication in e-health. However, mobile agents are also exposed to security threats, which are threats from malicious hosts and malicious agents [8],[9]. An agent carrying the information must be protected against other malicious agents, or malicious hosts that can tamper with its code or data. On the other hand, a host receiving the agent also concern on the incoming agent that might do malicious activities once its code is executed. To avoid these threats, cryptography protocols like encryption/decryption, digital signature, and hash code are used to protect mobile agent code and the carried message. The protocols are also able to provide mechanisms to verify the authenticity, confidentiality and the integrity of the code and data that arrived at the host. The generation of the symmetric keys for a communication between two users is according to the MLC approach [1], which will be described further in this paper.

The rest of the paper is organized like the following: Section II discusses the MLC approach. The security architecture is discussed in Section III, which explains the security requirements and the proposed security mechanism. The implementation is explained in Section IV. Then, the related work is discussed in Section V, and finally this paper is concluded with a summary in Section VI.

II. MULTILAYER COMMUNICATION APPROACH (MLC)

In order to fully understand our proposed security mechanism, the MLC approach is discussed in this section. The MLC approach is basically a security model that classifies information into five layers as described in Table I. The classification of the information is based on the sensitivity of the information being transmitted between two users. For simplicity purposes, the users in this approach are identified as Doctor, Patient, Nurse, Social Worker (SW), Paramedic, System Coordinator (SC), and System Administrator (SA).

In e-health, there is information that can be considered as extremely sensitive or less sensitive depending on where the information came from. For example, information that came

978-0-7695-3649-1/09 $25.00 © 2009 IEEE
DOI 10.1109/CNSR.2009.49

from communications between doctors, or a doctor and a patient, or a nurse and a patient, contains extremely sensitive data and must be protected with the highest security mechanisms. There is also information that is considered as less sensitive, for instance, a communication between a social worker and a nurse. This information can be protected with low security mechanisms. With the varying sensitivity levels of the information, the MLC approach is introduced, which classifies the information into five layers.

This approach is an alternative way to provide different kinds of communications to different users according to their needs. Different security mechanisms can be provided at each layer depending on the sensitivity of the data. Highest security mechanisms can be applied to the extremely sensitive information, while low security mechanisms can be applied to the low sensitive information. MLC treats every communication differently based on the sensitivity of the information being exchanged unlike Virtual Private Network (VPN) or Secure Shell (SSH). By providing different security mechanisms in every layer, an organization can choose its own security mechanisms flexibly, depending on cost and performance.

In MLC, each layer has its own range of key lengths for data and/or channel security. Data security involves the encryption, digital signature and hash of data, while channel encryption provides a secure SSL channel to transfer the data. For a communication between two people, say, Doctor and Nurse, the *layer of communication* (*com_layer*) is identified. The com_layer value is used to determine the security mechanisms and symmetric key lengths that will be applied to the data in a particular communication session. com_layer can be determined by using a *default layer* (L_0) value assigned to each user. For Patient, Doctor, and Nurse the L_0 value is Layer 1. For Paramedic and System Coordinator, the L_0 value is Layer 2. Social Worker is assigned to Layer 3, and finally System Administrator is assigned to Layer 4.

The following describes the rules to determine com_layer value for a communication between Sender and Recipient:

- If the L_0 values for Sender and Recipient are the same, then the com_layer for that communication will be Recipient's L_0
- If the L_0 value for Sender is greater than the Recipient's, then the com_layer for that communication is Sender's L_0
- If the L_0 value for Sender is smaller than the Recipient's, then the com_layer for that communication is Recipient's L_0

In summary, the com_layer value can be identified by comparing both L_0 values of Sender and Recipient. The one with a larger value will be chosen as com_layer. If both default layers are the same, that default layer will be used as the com_layer value. After the com_layer value is obtained, the security mechanisms for the communication can be determined, that is, whether the communication needs data security, or channel security, or both data and channel security.

TABLE I. THE CLASSIFICATION AND SECURITY MECHANISMS IN THE MLC APPROACH

Layer of communication	Types of data transmitted	Security Mechanisms and symmetric key lengths	
Layer 1: ***Extremely Sensitive data*** Doctor⇔Doctor Doctor⇔Patient Doctor⇔Nurse Nurse⇔Patient	Patient's personal information (name, address, age, gender, contact person etc), medical history, diagnosis, test result, current treatment and prescriptions	Data and channel security	192-bit key and longer
Layer 2: ***Highly sensitive data*** Paramedic⇔SC	Patient's personal information, medical information: allergic, blood pressure, current condition	Data security (using wireless network)	80-bit key to 191-bit key
Layer 3: ***Medium sensitive data*** Doctor⇔SW Nurse⇔SW	Patient's personal information, medication information	Channel security or data security	112-bit key to128-bit key
Layer 4: ***Low sensitive data*** SA ⇔ all users	Information on the application system, user account, non-medical related information such as IT technical problem	Channel security or data security	80-bit to less than 112-bit of key
Layer 5: ***No sensitive data or Public*** The public	*Open channel:* general information on the organization, information on health, diseases, frequently asked questions, annual reports, and services available	Secure open channel: ID and password	-
	Secure open channel: any user that wants to get access or contact information to any sensitive information (e.g.: a researcher)		

The com_layer is associated with the length of the symmetric key encryption algorithms:

Layer 1: key length =192-bit key and longer
Layer 2: key length = 80-bit key to 191-bit key
Layer 3: key length = 112-bit key to128-bit key
Layer 4: key length = 80 to less than 112-bit of key

The symmetric key is used for encryption processes to provide data security. Higher layer uses longer key length and lower layer uses shorter key length. The key lengths are also used as a guide to establish SSL channel to provide channel security. This is done by using a suitable cipher suite according to the layer. The next section describes the proposed security mechanism.

III. SECURITY ARCHITECTURE

A. Mobile agent paradigm

A mobile agent is a program that can migrate or move from a home platform to another, carrying its code and data. When migrating, the agent invokes a method for migration. For example, in Jade [10], a method called *doMove()* is used, which permits the agent to migrate from its home platform to a remote platform. Once migrated, the code will be executed in the remote platform and the original agent is destroyed. The remote platform is responsible to recreate the agent and allow it to execute once it arrives there. At the remote platform, the mobile agents may communicate with the agent in the remote platform to negotiate and ask for resources.

A user can delegate tasks to a mobile agent, dispatch the agent to another host, and execute the tasks there. In this research, the mobile agent is used to carry data to another host. There, the agent performs tasks on behalf of its owner. A mobile agent is robust, in a sense that if the destination platform is shut down while the agent is still there, the agent can take necessary actions such as to migrate back or terminate its activities [11]. It can send a notice to the home platform about its situation and terminate if required.

Fig. 1 describes a communication between two doctors, say Doctor A and B. Using the MLC approach, `DoctorA` agent obtains the com_layer value for the communication with `DoctorB` agent (which is Layer 1). After that, the message that will be transmitted will be encrypted using the symmetric key, which length is according to Layer 1 (192-bit key or longer). A token will also be encrypted and carried along with the message.

A `Mobile Agent` is created, and the code is signed by `DoctorA` agent, so that it could be verified at the Doctor B's host. The symmetric key will be kept by `DoctorA` agent until it is needed by the agent. An SSL tunnel is established (Layer 1 provide data and channel security), and `Mobile Agent` is dispatched to Doctor B's host to carry the message and its code (1).

After arriving at the host, the `Mobile Agent`'s code is verified and executed if proven to be valid. While at the remote platform, `Mobile Agent` can communicate with the home platform (2) by sending the token it carried to request for additional information to continue its execution. After receiving the token, `DoctorA` agent sends the information needed to `Mobile Agent` (3) so that the agent can continue with its tasks.

Another example is the communication between Paramedic and System Coordinator, which communicate using a mobile device. The com_layer value will be Layer 2, which provides symmetric key length from 80-bit to 191-bit to encrypt the message. Lower key lengths are provided to support low processing power devices. For Layer 3 communications, for instance, between a nurse and a social worker, the organization can choose for either channel security or data security. For channel security, an SSL tunnel is established based on the default cipher suite available. For data security, the key length for the symmetric key is between 112-bit to 128-bit. Layer 4 communications are similar to Layer 3, except for the key length, which is from 80-bit to 112-bit.

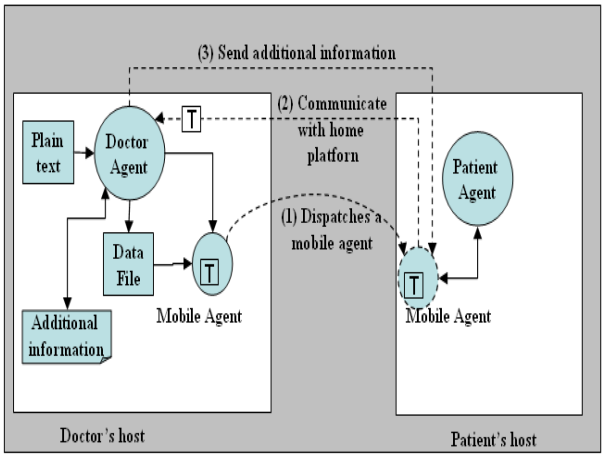

Figure 1. Communication between two users.

B. Control of the Data

In our approach, we focus on how Sender can securely transfer data to Recipient while maintaining control over the data. The 'control over the data' can be described as:

1. If the message carried by Sender Agent is seized by an attacker, the attacker still cannot recover the plaintext
2. Recipient or any other third party does not need to know the details of the decryption processes to recover the plaintext.

One way for Sender to gain control over the data, is to keep part of the requirements for the decryption processes secret, such as part of the agent's code, or parameters used for decryption. In this approach, the parameters for decrypting the ciphertext are kept with Sender until he/she knows that the mobile agent needs it. The parameters contain the symmetric key that encrypts the plaintext, hash of plaintext (for Recipient to check if the plaintext is tampered), and the information about the key, such as the types of the algorithms, key length, and encryption mode.

A token, which is an encrypted random number, is carried by the mobile agent to the Recipient's host. It is used as a 'phone home' [19] mechanism, where the agent sends back the token to the Sender. This is a way for the agent, to tell Sender that it wants the information kept at the Sender's side for the decryption processes.

The authenticity of the agent's code is verified once it arrives at the host. If the agent's code is valid, it is executed. Once executed, the agent is ready to perform the decryption process to recover the plaintext. However, the decryption key and the information of the key are at the Sender's side. This is where the agent sends the token back to Sender to get the necessary information. To illustrate this concept, the next section describes the communication protocols between two parties, Doctor and Patient, which involve a mobile agent.

C. The Communication Protocols

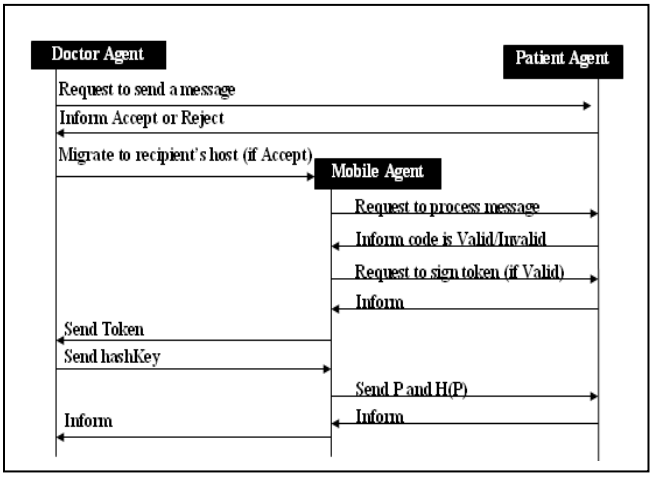

Figure 2. Communication Protocols between Doctor Agent, mobile agent, and Patient Agent

Doctor Agent creates an instance of a mobile agent and dispatches it to the Patient's host, carrying the encrypted plain text. At the Patient's host, the mobile agent is executed at the Patient's host environment. The mobile agent communicates with Patient Agent in the process to discover the plain text. The communication protocols between Doctor Agent (DA), Mobile Agent (MA) and Patient Agent (PA) are described in Fig. 2.

First, DA prepares the plain text, and applies appropriate security mechanisms to the plain text as well as to the MA's code. Then, DA sends a request to PA to send a message. PA replies with an 'accept' or 'reject' message, depending on whether DA is trusted by PA. If PA's reply is an 'accept', then, DA creates an instance of MA and dispatches it to carry the data to PA's host. Once arrived, MA initiates a communication with PA:

- MA requests PA to process the carried data
- PA processes the data
- PA informs MA whether the code is valid or not to be executed
- If the code is valid, MA retrieves the token and requests PA to sign it
- Once signed, MA sends the token back to DA
- DA processes the token, and if the token is not tampered, DA sends information for decryption process to MA.
- When the information is received, MA performs the decryption process to recover P
- Then, MA forward P and H(P) to PA for verification
- PA informs MA for the verification status
- Finally, MA informs DA the result and terminates.

The next sections discusses the security requirements at both Sender and Recipient's side.

D. Security Requirements

Before we explain the proposed security mechanism, we would like to discuss the security requirements to secure online communications between two users. We consider the previous example (Doctor-to-Patient communication) in Section III(C). Generally, both Doctor and Patient would want to make sure that the data transferred is safe from any unauthorized access, the data is not modified, and non-repudiation can be proven. These requirements are identified as confidentiality, integrity, and authenticity.

Confidentiality prevents an unauthorized user from accessing the information, which is usually accomplished by encrypting the data. Integrity prevents modification to the data. Any modification to the data will result in some changes in the chipertext. The changes can be detected by comparing two hashes of the data (data here means the plaintext or the mobile agent's code), one that is carried by the MA, and the other one, which is computed from the newly recovered data. If both are the same, then PA knows that the data is not tampered. Authenticity lets PA knows that the information is originally coming from DA. A digital signature can be used for this purpose. When PA verifies the signature using DA's public key, and found out that the signature is valid, PA knows that the signature has comes from DA, which it trusted.

To simplify the understanding of the concept, the following symbols will be used throughout this paper:

a. Public and Private keys of the recipient (PA): (*pubKr*, *privKr*)
b. Public and Private keys of the sender (DA): (*pubKs*, *privKs*)
c. Symmetric keys: *K1*, *K2*
d. public and secret key: (*Kp*, *Ks*)
e. Information kept at Sender's side: *info*
f. Plaintext, *P*
g. Hash of P, *H(P)*
h. Ciphertext, *C*
i. Signature, *S*
j. Agent's code: *Cd*
k. Hash of *Cd*, *H(Cd)*
l. A Random number *Rand*, Token, *T*

From the communication between Doctor and Patient, Doctor wants to send a plain text to Patient securely using mobile agent. Therefore, the plaint text as well as the mobile agent code that will be executed at the Patient's host must be protected. The following explains the security requirements at Doctor and Patient's side.

1) The security requirements at the Doctor's side:

a. Protection of the plaintext (*P*): *P* must be protected to avoid unauthorized access or modification from any third party. This is done by encrypting it with a symmetric key (*K1*). *K1* will be kept by DA and not sent to the PA's host until it is needed by MA.
b. A hash of *P*, *H(P)*: *H(P)* is created and sent to PA to verify that P is not modified.

c. Token, *T*: *T* is created to be carried by MA, which will be sent back to DA to get *K1*. K1 will be used to recover *P*.
d. Proof of identity: DA must make sure that PA will not fail to authenticate his/her agent at the Patient's host, by signing his/her agent's code using *privKs* to produce a digital signature *S*. If the agent fails to be authenticated, the agent will not be allowed to execute at the Patient's host.
e. Protection of the agent's code (*Cd*), *T*, and *S*: These three components must be protected from any unauthorized access. This is done by encrypting them with another symmetric key (*K2*). The signature needs to be encrypted because, an impostor, who is also trusted by PA can remove *S*, add his/her own signature, then take the agent's data, and add it to his/her own agent. Then this agent will be sent to PA without DA or PA knows that the agent is actually comes from the impostor. This is described as an attack on a targeted state [20].
f. A hash of *Cd*, which is *H(Cd)*, and send it to PA to verify that *Cd* is not modified.
g. Protection to the key (*K2*) and *H(Cd)*: This is to make sure that only PA can retrieve it. *pubKr* will be used to encrypt *K2* and *H(Cd)*.
h. Requires that PA provides his/her *privKr* to decrypt the message carried by MA, as well as to sign *T* before MA sends it back to the DA.

2) The security requirements at the Patient's side:

a. PA needs to check that the agent's code is indeed comes from DA. PA must be able to associate the identity of the MA arrives with its owner. This is done by using *pubKs* to verify the signature *S* against *Cd*.
b. It is important for PA to verify that the agent's code is not tampered before executing it. PA can compare a newly created hash of *Cd*, *H(Cd)* with the one carried by MA
c. After *P* is recovered through a decrypting process, PA also needs to verify that *P* is not modified or tampered, by comparing a newly created hash of *P*, *H(P)* with the one carried by MA.

After identifying the security requirements for both sides, the next sections onwards describe the step-by-step of the proposed security mechanism that happen at the Doctor's and Patient's host.

E. The Proposed Security Mechanism

This section discusses the cryptography protocols that are used to secure data and mobile agent's code. We refer to the previous example of Doctor and Patient communications (in Section III(C).

1) At the Doctor's host, DA does the following:

a. Identifies the com_layer value of the communication session. In this case, the com_layer value will follow rule number 1 (see Section III), which is Layer 1.

b. Generates two symmetric keys (*K1* and *K2*) according to the com_layer value. For this case, the key lengths for *K1* and *K2* will be in the range of 192-bit key and longer.

c. Encrypts *P* using *K1* to produce a ciphertext, *C*.
$$C = E(P)K1$$

d. Generates a random number *Rand*

e. Encrypts *Rand* with *K1* to generate a token, *T* that will be carried by the agent. *K1* is kept at the DA's side until DA receives *T* from MA.
$$T = E(Rand)K1$$

f. Generates public and secret key (*Kp*, *Ks*). After *T* is received, *Ks* is used to encrypt the information that is kept with DA for decryption process. *Kp* will be embedded in *Cd* and sent to the Patient's host. *Kp* will be used for decryption at the Patient's host. The generation of (*Kp*, *Ks*) is one time per communication session. These keys will be disposed once the communication session is over. This is to avoid any third party from using the key in the next communication sessions.

g. Signs *Cd* with *privKs* to produce a signature, *S*. The signature is used to verify that *Cd* is from DA.
$$S = E(Cd)privKs$$

h. Encrypts *Cd*, *S*, and *T* with *K2* to produce *Ciphercode*.
$$Ciphercode = E(Cd, S, T)K2$$

i. To allow only PA to retrieve *K2*, it is encrypted with *pubKr* together with the hash of *Cd*, *H(Cd)* to produce *Cipherkey*. TA can later compute a new *H(Cd)* from *Cd* in 8, and compare it with the one in *Cipherkey* to check whether *Cd* is valid and not violated.
$$Cipherkey = E(K2, H(Cd))pubKr$$

j. Saves *C*, *Ciphercode*, and *Cipherkey* in a file. Establishes SSL connection, and dispatch a mobile agent to send *Data file* to the Patient's host. The SSL connection is established to provide channel security. This is accordance with the MLC approach of Layer 1 that requires data protection and channel protection mechanism.

2) At the Patient's host, TA does the following: (described in Fig. 3)

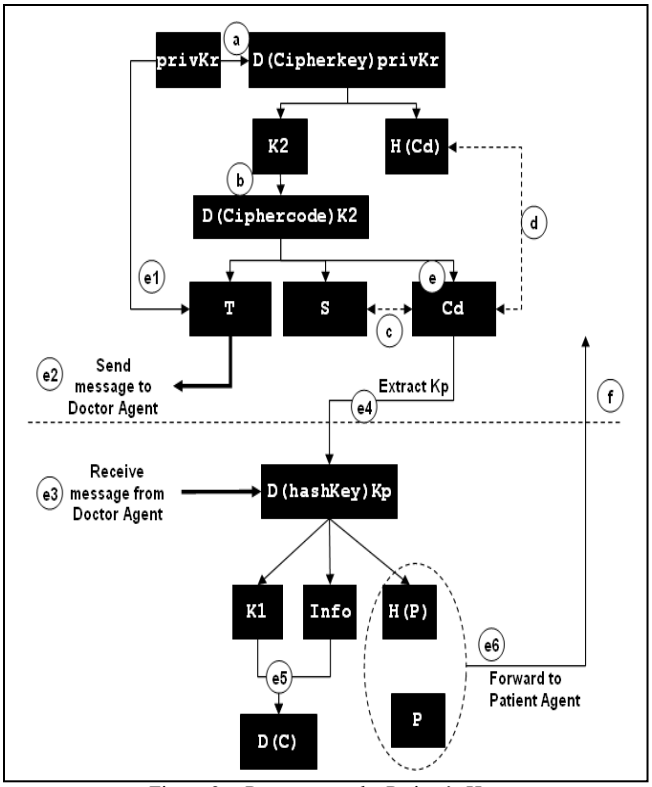

Figure 3. Processes at the Patient's Host

a. When PA receives a request to process a message from the mobile agent, PA uses *privKr* to decrypt *Cipherkey* to obtain *K2* and *H*(*Cd*)

D(Cipherkey) privKr = K2, H(Cd)

b. Then, *K2* is used to decrypt *Ciphercode* to get *Cd*, *S*, and *T*.

D(Ciphercode)K2 = Cd, S, T

c. Before executing Cd, PA checks that *Cd* is indeed come from DA. *S* is verified against *Cd* using *pubKs*

d. Calculate a new *H(Cd),* and compared it with the one in (1), to check if *Cd* has been tampered.

e. If both point (c) and (d) are valid, then *Cd* is executed.

e1. When *Cd* is executed, it is ready to decrypt *C* to get *P*. But first, *Cd* retrieves *T* and requests PA to sign it, using its *privKr*.

e2. Once signed, *T* is sent back to DA, to indicate that MA is ready for the decryption process

- Upon receiving *T*, DA verifies the signature and decrypts *T* with *K1*.
- If *T* is not modified, then DA encrypts *K1*, information on *K1* (*info*), and *H(P)* with *Ks* to produce *hashKey*. The corresponding *Kp* is stored within the code *Cd,* and will later be retrieved by *Cd* at Patient's host*:*

hashKey = E(K1,Info,H(P))Ks.

- Then, *hashKey* is sent back to the MA.

e3. *hashKey* is received from DA

e4. Again at the Patient's host, *Cd* retrieves *Kp*, and use it to decrypt *hashKey* to obtain *K1*, *info*, and *H(P)*.

e5. *Cd* uses *K1* and *info* to decrypt C to get plain text, *P*.

e6. Afterwards, both *P* and *H(P)* will be forwarded to PA for verifying purposes.

f. Finally, PA verifies *P* by calculating a new *H(P)* from *P*, and compare it with the one forwarded earlier. If proved valid, keep *P*.

By using this mechanism, DA has the advantages of gaining control on the data carried by MA, because PA or any third party does not know about the details of the encryption key. Even though an attacker could get hold of the message from DA (*hashKey* at point e3), the attacker still cannot recover the plaintext as the key to decrypt *hashKey*, which is *Kp*, is at the Patient's host. In addition, *Kp* is disposable, which is created only once per every communication. *Kp* and the corresponding *Ks* will be removed once a communication session is terminated.

Another advantage for DA is that, when the token is received, DA knows that PA has been correctly executing MA:

- The agent code will only be executed if PA has approved the authenticity and the integrity of the code.
- The access to the resources at the Patient's host is not denied. In this case, PA must provide its private key to decrypt and sign specific message (in a safe way through a specific function that will not risk the Recipient's host).

For PA on the other hand, it does not have to be burdened with the details of the decryption of the plaintext. PA only require to verify that the agent is indeed comes from the Doctor's host by checking the signature and the integrity of the agent's code. If both are valid, PA knows that the message and agent are from the trusted sender. In addition, PA can check whether the plain text is not tampered by calculating a new hash code, and compare it with the one received from the agent.

IV. IMPLEMENTATION

We have implemented the proposed security mechanisms described in this paper as a proof-of-concept, using Jade [18], a Java-based FIPA compliant agent platform. Each user's machine run a Jade agent platform and each agent on the platform is able to accept foreign agent (which is the mobile agent) by enabling the inter-platform mobility service. Each agent is created with behaviours that represent the roles or tasks of the agent. For our research, we implement agent classes that represent Sender such as DA (`SenderAgent`), Recipient such as TA (`RecipientAgent`), and the mobile agent such as MA (`MobileAgentData`). `SenderAgent` and `RecipientAgent` are instantiated by every platform, so that each platform can act as Sender and Recipient. `MobileAgentData` will be created by `SenderAgent` to carry the data to the Recipient's side. `MobileAgentData` contains a behaviour instance that implements *Cd*. *Cd* will be executed after being verified by `RecipientAgent`. We use the FIPA-ACL performatives and ontology for the agents' communications. At the Sender's side, `SenderAgent` is responsible to prepare messages such as described in Section III-D(A). At the recipient's side, `RecipientAgent` and `MobileAgentData` are responsible to execute the steps in Section III-D(B).

For a proof-of-concept, we have implemented a one-to-one communication for each layer using PKI support. As for the Doctor-Patient example, we provide AES 256-bit of key (or alternatively, we can use AES 192-bit of key for Layer 1) for both symmetric key. SHA1 algorithm is used to create hashes of *P* and *Cd*. For the asymmetric key, we create a pair of RSA key on the run, where the public key is embedded into the agent's code. The public key will later be retrieved by `MobileAgentData` to decrypt the information received from the Sender's side.

For Layer 2, we provide Blowfish 80-bit (or Blowfish-112 bit). Layer 3 and 4 uses channel or data security. For channel security, we use the default SSL connection with the default SSL configuration [13] provided by Jade. Alternatively, for data security, we use AES 128-bit (or Blowfish 112-bit) for Layer 3, and for Layer 4, we use Blowfish 80-bit (or Blowfish 112-bit).

Because we have only implemented this protocol in one-to-one communication, we have not taken into account the key management, especially the symmetric key management, which will be our future work.

V. RELATED WORK

The use of mobile agents in e-health has become a well accepted paradigm to support online communications [14],[15],[16]. There are many researches that use mobile agent approach to carry sensitive data from Sender to Recipient over the network. As the data is sensitive, mechanisms to protect the code and data are developed. This section discusses some of the solutions of mobile agent security that based on the cryptography protocols

A mobile agent security mechanism is proposed in [16],[17] where the agent itself carries the protection mechanisms without depending on the Sender/owner's platform. At the destination host, the mobile agent code is authenticated and the code requests a service from the platform to decrypt the message. To avoid malicious code injection to the data carried by the agent, the mobile agent carries its own protection by using disposable key pair (public and private keys). The Sender's agent signs the data using the private key. The corresponding public key is sent to the Recipient's host together with the signature, which is encrypted with the Recipient's public key. The Recipient's agent can check the integrity of the data by verifying the signature with the public key.

There are also mechanisms that provide Sender a method to detect modifications on the mobile code executed at the remote hosts. A tracing mechanism is presented in [18], which consist of a unique identifier of a statement and a signature. When the targeted platform has finished executing a code, it produced a tracing log. Then, the result of the tracing is sent back to the owner of the agent so that the owner can make sure that the code has been executed correctly. The 'phone home' approach [19] describes a method for a mobile agent to contact Sender, to tell Sender the current state it was in, or to transfer important data home. Therefore, any tampering with the agent can be detected by Sender by checking the data received from the agent. Further reading on mobile agent security can be found in [20],[21] that summarize research works on threats to the agents and platforms, as well as the countermeasures.

In our approach, we keep the encryption details at the Sender's side. Therefore, Recipient or any other third party does not need to know the details of the decryption process. For that reason, it is futile for any third party that tries to peek into the data in transit, because the encryption details are not there. In addition, the dependency of the mobile agent to the Sender's platform for the additional information for encryption processes, allows Sender to know that the agent has been correctly executed at the Recipient's platform.

VI. CONCLUSION

A security mechanism for mobile agents based on our previous research results is presented in this paper. The mobile agent is used as a supporting tool to carry sensitive data in e-health. Cryptographic protocols are used to secure the mobile agent and the carried data. The key to decrypt the ciphertext is kept with Sender. A token is carried by the mobile agent to the Recipient's host. There, the token is sent back to Sender as a sign that the agent needs the key to decrypt the ciphertext. By using this mechanism, Sender can gain control over the plaintext, because Recipient or any other third party does not know the details of the decryption processes. On the other hand, Recipient does not have to cater for the decryption processes and only concern on authenticating the agent is from the trusted source, and verifying the integrity of the code and plain text.

ACKNOWLEDGMENT

Rossilawati Sulaiman is funded by a joint scholarship from Universiti Kebangsaan Malaysia (the National

University of Malaysia) and the Ministry of Higher Education, Malaysia.

REFERENCES

[1] Sulaiman, R., Sharma, D., Ma, W., & Tran, D. (2008). A Security Architecture for e-Health Services. Paper presented at the The 10th International Conference on Advanced Communication Technology (ICACT), Phoenis Park, Korea.

[2] Eysenbach, G. (2001). What is e-health? J Med Internet Res, 3(2)from http://www.jmir.org/2001/2/e20/

[3] Mitchell, J. (1999). From telehealth to E-health: The unstoppable rise of E-healthJohn Mitchell & Associates for the Federal Australian Department of Communications, Information Technology and the Arts (DOCITA).

[4] Shou, L. L., Gingrich, D., Lewis, P. R., Mauger, D. T., & George, J. H. (2005). Enhancing doctor-patient communication using email: A pilot study. The Journal of the American Board of Family Practice, 18, 180-188.

[5] Pagliari, C., Donnan, P., Morrison, J., Ricketts, I., Gregor, P., & Sullivan, F. (2005). Adoption and perception of electronic clinical communications in scotland. Informatics in Primary Care, 13(2), 97-104(8).

[6] Tang, P., Black, W., Buchanan, J., Young, C., Hooper, D., Lane, S., et al. (2003). PAMFOnline: Integrating EHealth with an electronic medical record system. Proceedings of the American Medical Informatics Association Annual Symposium, 649-653.

[7] J. White, "Mobile Agent White Papers", General Magic, 1996.

[8] Roth, V. (2002). On the Robustness of Some Cryptographic Protocols for Mobile Agent Protection. Paper presented at the Proceedings of the 5th International Conference on Mobile Agents.

[9] Chess David M. Security issues in mobile code systems. In Mobile Agents and Security, volume 1419, pages 1-14. Springer Verlag, 1998.

[10] Jade, Java Agent Development Framework. http://jade.cselt.it, 2006.

[11] Danny, B. L., & Mitsuru, O. (1999). Seven good reasons for mobile agents (Vol. 42, pp. 88-89): ACM.

[12] Roth, V. (2002). On the Robustness of Some Cryptographic Protocols for Mobile Agent Protection. Paper presented at the Proceedings of the 5th International Conference on Mobile Agents.

[13] Exposito, J A., Ametller, J., Robles, S. (2003), lHow to use the new HTTP MTP with JADE. http://jade.tilab.com/doc/tutorials/HTTP_UAB.html

[14] Pitsillides, A., Samaras, G., Pitsillides, B., Georgiades, D., Andreou, P., & E.Christodoulou. (2006). Virtual collaborative healthcare teams for home healthcare. Journal of Mobile Multimedia (JMM), special issue on Advanced Mobile Technologies for Health Care Applications, 2(1), 023-036.

[15] Germanakos P., Mourlas C., & Samaras G. "A Mobile Agent Approach for Ubiquitous and Personalized eHealth Information Systems." Proceedings of the Workshop on 'Personalization for e-Health' of the 10th International Conference on User Modeling (UM'05). Edinburgh, July 29, 2005, pp. 67-70.

[16] Pedro Manuel, V.-M., R. Sergi, et al. (2006). Secure Integration of Distributed Medical Data Using Mobile Agents, IEEE Educational Activities Department. 21: 47-54.

[17] Ametller, J., Robles, S., & Ortega-Ruiz, J. A. (2004). Self-Protected Mobile Agents. Paper presented at the Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1.

[18] Vigna, G. (1997). Protecting Mobile Agents through Tracing. The 3rd ECOOP Workshop on Mobile Object Systems.

[19] Michael, J. G., & Brian, D. M. (1999). Protecting the integrity of agents: an exploration into letting agents loose in an unpredictable world (Vol. 5, pp. 10-17): ACM.

[20] W. Jansen and T. Karygiannis, NIST special publication 800-19 - mobile agent security, National Institute of Standards and Technology. URL http://csrc.ncsl.nist.gov/mobilesecurity/Publications/sp800-19.pdf

[21] N. Borselius, "Mobile agent security", Electronics and Communication Engineering Journal, IEE Press, Vol. 14, No. 5, 2002.